# Considerations Regarding the Cross-Platform Mobile Application Development Process

Marius POPA
Department of Economic Informatics and Cybernetics
Bucharest University of Economic Studies, Romania
marius.popa@ase.ro

*The huge demand for mobile applications is caused by the growing share of the smart hand held devices. Covering that demand with high quality mobile applications developed, tested and released faster than traditional methods requires new development process approaches and tools having superior capabilities to achieve the developer's purposes. The paper offers a comprehensive introduction to mobile applications, mobile development strategies, building environments and process, examining the particularities of cross-platform mobile application development lifecycle. The cross-platform approaching gives the mobile software developer the benefits of a larger market share and more potential users of the idea behind the mobile application functionality.*
***Keywords:*** *Mobile Application, Cross-Platform, Mobile Development*

## 1 Mobile Applications

Mobile applications are computer programs specific to low power devices such as smartphones, tablets, mobile phones, personal digital assistants and other handheld devices. The most part of such devices are ARM-based architectures. ARM is an acronym for Advanced RISC Machine and describes a RISC-based architecture of the processor.

Mobile applications are executed in the native machine format under the supervision of execution environments such as Android, BlackBerry, iOS, Symbian OS, Windows Mobile, Windows Phone as the most known and used operating systems on handheld devices. Each platform has its own hardware, security issues and unique features.

For the previous execution environments, there is a wide range of programming languages integrated in development environments (IDEs) to produce software able to be executed on mobile devices running various operating systems. The programming languages and IDEs address specific features of the execution environments running on mobile devices. Many mobile execution environments provide free Software Development Kits (SDKs) and IDEs to developers. Also, the mobile application developers can use third-party IDEs. Part of such IDEs offer cross-platform development capabilities.

The most relevant feature of a handheld device is mobility. As a consequence, the availability and usability are most important features of an application running on such devices. The fulfillment of the two features depends on the skills and capabilities of the development team regarding the mobile operating system, programming languages, connectivity models to Internet and hardware features of the devices. Deployment of a mobile application on as many devices as possible means covering the availability and usability features by the development team. The mobile application developers must be aware of the difference between the following particular issues of such application development as it states [1]: mobile operating systems, cross-platform development, variety of screen resolutions, device sensors, touch screens, mobile standards, security issues and production and deployment models.

The strategies for mobile application development are:

- *Native applications* – they are platform-dependent applications developed with SDKs and tools provided by platform owner; the strategy restricts the number of mobile devices running the application, but it has a big advantage regarding

optimization and using of everything the platform offers to developers; there are mobile applications for which the native strategy is the single choice;

- *Web applications* – they are the applications developed using the standard web technologies: HTML, JavaScript and CSS; the advantages of this strategy are: large spreading of web technologies among the software developers, reduced development costs from native strategy, single codebase updatable anytime, permanent availability in the Internet and avoidance of the application stores with specific rules for application publishing; web application cannot use the device capabilities and offline use is more constrained [6];
- *Hybrid applications* – they are web application wrapped in a native shell; they mitigate the disadvantages of web applications providing access to mobile platform features through the native shell; access between the web specific programming language and the native code is made by a bridge providing access to the device capabilities; hybrid strategy combines the advantages of native and web strategies; it is the best strategy for cross-platforms application development and its advantages are stated at [10] and aims:
  - Single time coding and deployment for various platforms;
  - Native calls from native shell;
  - Application availability in offline mode;
  - Providing to a large number of users;
  - Local processing is possible;
  - Distribution through application store;
  - Providing notification about updates.

At [10], a Cross Platform Architecture and Code Sharing implementing the hybrid strategy in Windows Phone are depicted, Figure 1.

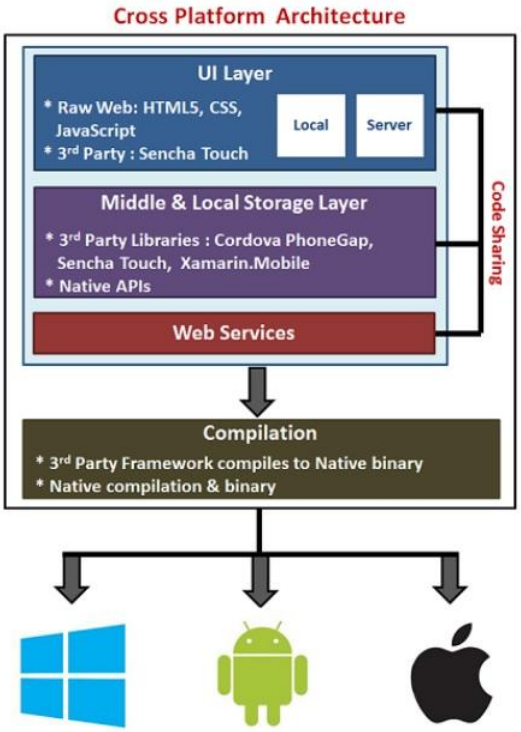


**Fig. 1.** Cross Platform Architecture & Code Sharing [10]

A new trend regarding the software development is stated by replacement of IDE by Integrated Cloud Environment (ICE) [6]. Using ICE, a developer can access multiple platforms, features and tools to build hybrid mobile applications. Local installation, recompilation, deployment on mobile devices are avoided using cloud services.

In European Union (EU), the availability and development of cloud services will increase as the EU's strategy called Europe 2020 is implemented. Europe 2020 is the EU's growth strategy for the coming decade. The strategy stimulates the intelligent growing on three major initiatives. The first initiative is Europe's Digital Agenda having the following goals [14]:

- Digital Single Market;
- Interoperability and Standards;
- Trust and Security;
- Fast and ultra-fast Internet access;
- Research and innovation;
- Enhancing digital literacy, skills and inclusion;
- ICT-enabled benefits for EU society.

The above aspects are reasons to consider the using of ICE in the future instead of IDE for

software development and mobile applications in particular.
The ICE supporters consider Cloud Development as a solution of the problems introduced by Desktop Development. According to [21], the reasons are:

- Complicated configuration management is time consuming and a challenge for software developers who are transformed in part-time system administrators;
- Decreased productivity because of slower machine with insufficient hardware resources, especially memory and disk, for the IDE used on desktop;
- Limited accessibility is generated by lack of access from other devices than the desktop;
- Poor collaboration because the IDEs have not communication and collaboration components.

ICE is a centralized architecture, able to be shared among software developers in order to co-edit, co-build and co-debug [21]. Each software developer has memory and computer resources and a configurable workspace required by development tasks.
However, there are some issues regarding the moving from local to cloud software development:

- Availability and speed of Internet access;
- Costs of cloud development services;
- Less control and flexibility of the development environment.

Currently, transition from IDE to ICE appears to be a long time process. However, increasing the using of cloud services will change the way applications are designed, developed, tested and deployed, leading to building of new development environments.

## 2 Frameworks Used in Cross-Platform Mobile Application Development

As it was highlighted in the previous chapter, there are more strategies can be used in mobile application development. For each strategy, there is development frameworks provided by execution environment owner or third party vendors.
Mobile web applications are web applications built for handheld devices like smartphones, tablets and so forth accessed through the web browser application installed by operating system of the device. Mobile native applications are built for a particular device having particular hardware specifications and operating system. Mobile native applications are implemented in different programming languages like Java, Objective C, C# and so forth. Building such an application needs knowledge of various programming language by developer. The programming language issue is removed by cross-platform frameworks for mobile application development.
Cross-platform frameworks accelerate the mobile application development and cheapen it. The reason is that a mobile application for different platforms is built with a single development process.
According to IDC Worldwide Mobile Phone Tracker [15], the mobile operating system market share highlights the dominance of two vendors as it is shown in Figure 2.

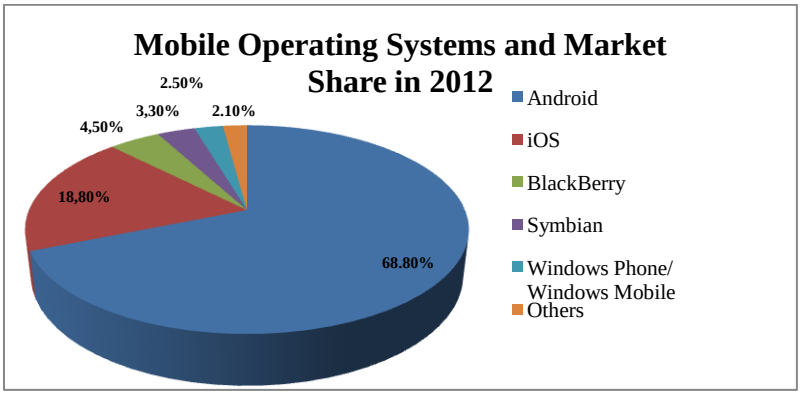


**Fig. 2.** Mobile operating systems and market share in 2012

Some of the most known tools for mobile application development are presented below.

***Appcelerator*** is an open source framework supporting Application Programming Interfaces (APIs) for Android, BlackBerry and iOS.

The Appselerator framework is depicted in Figure 3.


**Fig. 3.** Appcelerator framework architecture, from [8]

The features ensured by the components of the platform are [8]:

- Building of native applications – the prototype of the application is built using *Appcelerator Studio*; the layout, style and business logic of the application are defined using *Appcelerator Alloy*;
- Integration of extensions – third party modules, developers and partners community are included in *Appcelerator Marketplace* component;
- Connection to data – the application is connected to mobile cloud through scalable service provided by *Appcelerator Cloud* pre-built services component; also, *Appcelerator Cloud* provides custom services for data integration with back end systems;
- Higher quality of mobile application in shorter time – debugging is made on-device in *Appcelerator Studio* to identify the development issues; testing is automatically made using *Appcelerator Test* to ensure the highest quality and stability;
- Analyzing the usage and performance – usage is analyzed by *Appcelerator Analytics* to discover usage ways and trends as starting point for mobile application improvements; application performance is established by *Appcelerator Performance Management* component which captures information regarding exceptions, application crashes, memory and CPU use when an event is triggered;

The framework provides support for reusable widgets and component building in JavaScript. The benefits aim [8]:

- Application development life cycle much shorter than traditional methods;
- Easy maintenance of the code;
- Reusing of the widgets;
- Prototypes are created in days instead of months;
- The entire prototype is included in the final mobile application.

***RhoMobile*** is an open source Ruby-based platform used for a wide range of mobile device and operating systems, including Android, BlackBerry, iOS, Symbian, Windows Mobile and Windows Phone. RhoMobile Suite framework includes 3 components [19]:

- *RhoConnect* – connects mobile applications to data source, including enterprise back-end Customer Relationship Management (CRM), Enterprise Resource Planning (ERP) systems, web services and databases; the features of the component includes:
  - Data synchronization improvements – data tracking on each device, automatic and transparent data identification and synchronization, minimal data exchange during synchronization;
  - Zero-touch integration – made by framework plug-ins which create the appropriate data connections;
  - Backend changes management – adding new information in backend systems without modifying the mobile application;
  - Fast and easy scaling – "NoSQL" approach to data storage, using

available hardware and space no matter where they are located;

- *RhoElements* – is designed to enterprise mobility offering to the employees the information they need no matter what they use the wireless network or not; the hybrid and native HTML5 mobile applications developed by the component have the following features:
  - Offline Cache capabilities operating in an occasionally disconnected model;
  - Compiled byte code for a better performance;
  - SQLite support for offline capabilities;
  - Device integration capabilities fully enabled;
- *RhoStudio* – offers tools to focus on developing application features instead of different platforms with particular SDKs, testing and debugging tools, application version management; the component includes simulation tools for application running on different platforms and it has the following features: Macintosh and Windows platforms for mobile application development, simulator, remote debugging, code inspector and profiling.

***PhoneGap*** is a mobile development framework written in JavaScript, HTML5, CSS3, Java, C++, C# and Objective-C. Mobile applications built by PhoneGap are hybrid with layout rendering via web views instead of native User Interface (UI) framework and a core native code wrapped by the web component ensuring the access to the native device APIs.

PhoneGap platform enables features in mobile application development for the following mobile operating systems: Android, Bada, BlackBerry, iOS, Symbian, Tizen, webOS and Windows Phone. It is based on open-source Cordova project that allows the access to the native functionalities from JavaScript and the result is rendered by the web technologies.

The PhoneGap architecture is depicted in Figure 4.

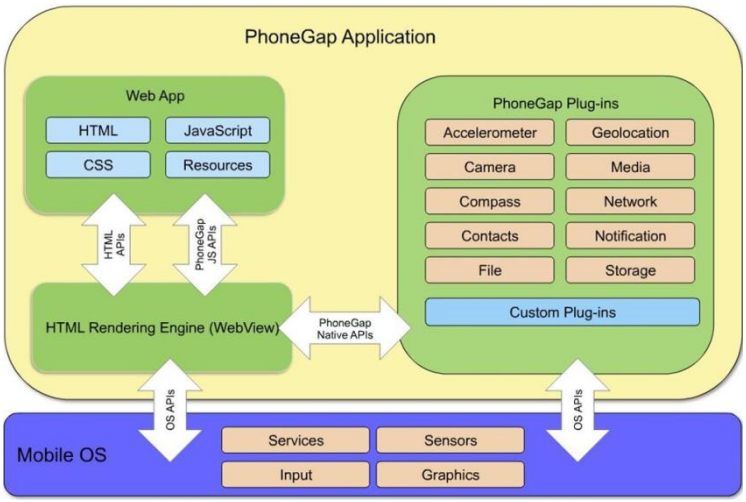


**Fig. 4.** PhoneGap framework architecture, from [17]

The PhoneGap components have the following roles within framework architecture [20]:

- *WebApp* – uses web technologies like HTML, CSS and JavaScript to build the UI layer of the mobile application;
- *PhoneGap Plug-ins* – enable the access to the native operating system functionalities using JavaScript as technology used for application logic; mobile software developers can write native plugins together with JavaScript interfaces to be used for PhoneGap applications;
- *HTML Rendering Engine* – uses the web view of the native mobile operating system for layout rendering.

The final mobile application developed by PhoneGap is distributed in a binary format after a packaging process in accordance to

the politics of the mobile operating system owners. The packaged application is in the same format with the native applications sold through distribution channels of the mobile application vendors.

The PhoneGap mobile application acts like a client communicating with a server application. The server application handles the client application logic and data management in a separate data repository, normally a relation database.

The client application uses standard web technologies for communication with the server application (Apache, IIS, etc.) upon HTTP requests, REST-ful XML services, JSON services, or SOAP. It uses single-page application model which is memory persistent. The page content is updated via HTML DOM, data is retrieved via server application using AJAX techniques, and variables are memory persistent thanks to JavaScript technology [20].

On top of the PhoneGap framework, there is a cloud service called PhoneGap Build enabling the mobile application development in the cloud.

***Sencha Touch*** is a mobile application development framework which enables HTML5 based mobile application development. The device hardware components are accessed by Sencha Touch through a native shell. It builds mobile applications running in web browsers like Android browser, Google Chrome for Android, BlackBerry 10, Bada Mobile Browser, Kindle Fire Browser, Windows Phone 8 and Windows 8 IE10, Mobile Safari and Tizen browser. The mobile applications built with Sencha Touch run in mobile browser with a native-app-like experience for the final user [22].

To build cross-platform mobile application with Sencha Touch, the software developers need the following tools [12]:

- Sencha Touch SDK and SDK Tools – are free on Sencha web-site;
- A web server running on the local machine;
- A modern web browser.

In Sencha Touch, the Graphical User Interface (GUI) controls are optimized for touch input and they are called components. The components can be themed according to the target device, using a style-sheet language interpreted into CSS and called Syntactically Awesome Stylesheets (SAAS).

Features of the Sencha Touch aim the following elements [9]:

- The learning curve is very steep – the architecture and components are inherited from Sencha Ext JS and require a lot of time for learning;
- The object-oriented approach – there are rules to access the native features;
- Extensibility – the framework is not extensible with customized plug-ins by software developers;
- Component creating – it is more difficult than in other frameworks; it must follow the Sencha GUI controls;
- JavaScript dependence – the UI is built in only in JavaScript; any update supposes JavaScript knowledge;
- Ready-made components library – it is included in Sencha Touch for a fast and easy building of UI.

Sencha Touch has the ability to dynamically load classes when they are needed, having many benefits in development and production.

***Xamarin*** is a cross-platform development framework to build native mobile applications written in C# for Android, iOS and Windows Phone. The Xamarin platform includes elements for Android and iOS application development, as they are presented in [13]:

- Programming language – it is C# with familiar syntax and advanced features;
- Mono .NET framework – provides a cross-platform implementation;
- Compiler – builds native applications or integrated .NET application and runtime;
- IDE tools – builds and deploys Xamarin projects.

Building cross-platform mobile applications with Xamarin must consider the following key points as [13] presents:

- Code writing in C# – the C# code is used on Windows Phone and it is ported to Android and iOS by Xamarin platform;

- Using the MVC design pattern – the application is architected using Model/View/Controller approach used for UI development;
- Building native UIs – each mobile operating system has a specific UI layer like MonoTouch.UIKit, Android, XAML/Silverlight and Metro.

Reusable code is separated into a Core Library to maximize the code sharing across platforms, as it is shown in Figure 5.

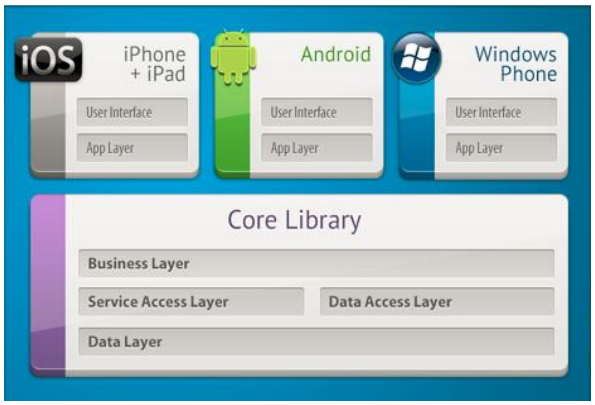


**Fig. 5.** Separation of reusable code into Core Library, from [13]

The Xamarin limitations aim the following issues, as they are stated at [23]:

- UI code is differently used and implemented for each mobile operating system (Android, iOS); this issue introduce a lower productivity of the designers and developers, but a smaller complexity and a better user experience;
- Xamarin framework does not allow developing of the reusable components in other projects for native or HTML5 mobile application development;
- Poor available support for mobile developers who search issues on web having a negative impact on productivity;
- Specific knowledge about the Xamarin framework increasing the time assigned to learning instead the programming language and native APIs;
- Large or complex components added to the mobile application increasing the occurrence of bugs in the final application.

Xamarin is a powerful framework in cross-platform mobile application development because it can use as a Visual Studio plug-in by .NET programmers to write mobile applications in C# for different platforms like Android and iOS.

***jQuery Mobile*** is a cross-platform development framework using HTML5 for mobile applications available on Android, Bada, BlackBerry, iOS, MeeGo, Symbian, webOS and Windows Phone. It is a unified UI system based on jQuery and jQuery UI for touchscreen devices.

The key features of jQuery Mobile are stated at [16] and include the following elements:

- jQuery core – is used to build jQuery Mobile having a familiar and consistent syntax and an easy-to-know and use of UI code and patterns;
- Compatibility – ensures a large coverage of mobile operating systems and web browsers;
- Lightweight – the code size and dependencies are small with benefits for application speed;
- Modularity – the application includes optimize modules with implemented functionalities;
- HTML5 Markup-driven configuration – is used for fast development and less scripting;
- Progressive approaching – the enhancements aim the core content and functionality of all mobile platforms, and a similar experience with installed applications on the newer mobile operating systems;
- Responsive design – aims the automatic scalability of the application to the hardware specifications of a particular mobile device;
- Powerful navigation system – is based on Ajax as exchanging data technology to minimize the Internet traffic for UI loading on the mobile device;
- Accessibility – is a feature to facilitate that the mobile application works for assistive technologies like text-to-speech, speech-to-text and so forth;
- Input support – both touch screen and mouse events are considered as input methods;

- Unified UI widgets – aim the enhancement of native controls with optimized operations;
- Theming framework – is a component to make applications easy to build at UI level;
- Speed download customization – specific modules of jQuery can be downloaded with Download Builder component of jQuery Mobile.

The mobile applications developed with jQuery Mobile are web applications with HTML5 as markup-driven language. Inside the page code, some conventions and rules are followed for a functional mobile application.

jQuery Mobile solves the problem of plenty of mobile devices and platforms by using the markup language interpreted by mobile web browser in order to adjust the application functionalities to the host mobile platform. Also, jQuery Mobile offers software developers the tools to create multiplatform and customized mobile applications easily.

The cross-platform development frameworks solve the issues of mobile application portability across different mobile platforms. There are several ways to choose the mobile application development appropriated to the developer's goals, taking into consideration the type of mobile application and the targeted quality and distribution of it.

## 3 Mobile Application Development Process

Mobile application development process has to take into consideration the wide range of mobile devices with different mobile operating systems or versions of them. Each mobile operating system or OS version has particular features like hardware, security, screen sizes and screen resolutions, connectivity models and speeds, and other capabilities. Development process considers the above features to affect the behavior and performance of the mobile applications.

As consequence, the mobile application developers must be prepared for:

- Coding the mobile application for different mobile operating systems or OS versions;
- Implementing the mobile application for a large range of screen resolutions;
- Considering the specific hardware features of the device which can give special functionalities to the application as against the others.

Compared to traditional application development, there are some steps that are different during the mobile application development process. These steps aim testing and quality assurance, support, and deployment and distribution [1].

According to [1], the mobile application development process is preceded by:

- Strong demand – a critical mass of hardware devices can provide a sufficient demand of mobile applications; this issue is critical to provide a significant return of investment;
- A solid foundation – there are some mobile applications, services and platforms as starting point in future mobile application development;
- Mobile device management – it is used for service delivering and push security policies instead of old containerized service delivering;
- Service-oriented architecture (SOA) – an abstracted service layer is used by mobile application to access the business logic and data;
- Adequate security – it aims data encryption web portal gateways for mobile devices.

The above issues are considered before starting the mobile development process for integration of mobile devices into the enterprise architecture.

In context of integration of mobile devices into the enterprise architecture, the mobile application development process has two stages [1]:

1. Establishing the software candidate for mobile development; the activity is based on a decision matrix;
2. The way of mobile application delivering.

In the first stage, the decision matrix development considers the following items [1]:

- Case – the project team examines the candidate software for mobile deployment through its functionalities and user experience; usually, the applications that enable small tasks or key updates are performed within a short time period are best candidates for mobile deployment;
- Consumer – the project team determines the user base and target platform; if users base is substantial, then the mobile application is developed for one or more target platform; also, a single platform can be used in the enterprise environment limiting the mobile devices to a single one;
- Content – the following issues are determined by the project team:
  - Existing services facilitating the mobile devices integration into the enterprise architecture;
  - Performance of mobile device and user experience after mobile application deployment;
- Scope and cost – the project team considers the project scope expanding and additional cost of SOA, possible multiple interfaces and team training.

The second stage establishes the deployment mechanism of mobile application. There are two possible mobile application deployment mechanisms [1]:

- Platform-independent – includes web and hybrid mobile applications;
- Platform-dependent – uses Mobile Enterprise Application Platform and a virtualized mechanism.

The elements necessary to build a mobile application development framework are [1]:

- Guidance documentation – offers information both to project managers and software developers regarding the differences between a desktop applications and mobile applications;
- Enabling capabilities – regards the right operating of the network resources, allowing the mobile applications to connect to enterprise interfaces and services;
- Supporting resources – aim all those who can provide help for mobile application development process: mobile developers, third party suppliers, engineering and so forth.

Mobile application development process has to consider the particular features of the mobile devices having a wide range of hardware specifications and operating systems. As effect, each mobile platform introduces particular considerations in mobile development. Also, the mobile application development lifecycle does not differ very much from traditional software development lifecycle so some common considerations about cross-platform mobile development can be stated.

Cross-platform mobile development common considerations are stated in [7] and they regard:

- Multitasking – mobile developers faces two challenges:
  - The limited screen size induces displaying a single or few applications running simultaneously;
  - Multiple applications running simultaneously require battery power; therefore, multitasking reduces the life time of the battery power and a shorter time for mobile device using;
- Form factor – mobile operating systems run on different devices having various hardware specifications; the form factor considers the wide range of the screen size across the mobile devices; the UI controls of the mobile application are designed to be effective on smaller screen sizes;
- Device and mobile operating system fragmentation – consider the wide range of mobile devices having different hardware specifications; the following elements have to take into account during the cross-platform mobile development lifecycle:
  - Conceptualization and Planning – mobile application development has to consider that some hardware specifications lack on some devices; therefore, the mobile application must treat the exception cases when it cannot work with different hardware features on certain mobile devices;
  - Design – screen sizes and resolutions

are key factors for an effective User Experience (UX) and User Interface (UI) design;
  - Development – before using a hardware feature from the application code, the mobile operating system is asked to report whether the device feature is present on it; after that, it follows the working operations with that hardware feature;
  - Testing – is made early in mobile development lifecycle; also, mobile devices having the same hardware specifications can have different behaviors; therefore, testing has to make on actual mobile devices;
- Limited resources – mobile devices still have limited capabilities as against the computers; the device performance depends on the complexity of the mobile application uses the device processor intensively.

The cross-platform mobile application development lifecycle is not very different from the desktop development lifecycle, but it must consider the particular specifications of the mobile devices as hardware and software features. A version of mobile application development lifecycle is stated in [7] and it has the following stages:

1. ***Inception*** – represents the starting point in mobile application development; it contains the idea regarding what the application will do refining in solid specifications for the next stages; the success of the mobile application is assured whether the following considerations are regarded:
   - Competitive advantage – established by similar mobile applications already deployed and the differences between these applications and application to be developed;
   - Infrastructure integration – aims the existing enterprise infrastructure being used to integrate or to be extended by the future mobile application;
   - Value – states why the users are interested by the future application and how they will use it;
   - Form/Mobility – states how the future mobile application will work in a mobile form factor and what is the added value bringing by the mobile technologies;
2. ***Design*** – defines the UX and turning the UX into the UI; user experience design aims the general layout of the application, how the application works and so forth; defining the UX aims:
   - the Interface Guidelines offered by the mobile operating systems providers have to take into consideration; for example, the mobile platforms implement switching between the application sections differently: tab bar at the bottom of the screen (iOS), tab bar at the top of the screen (Android), Panorama view (Windows Phone 7);
   - hardware features of the mobile device affect the UX; for example, iOS mobile devices have not the Back button; therefore, the navigation is implemented in software;
   - form factor affects the UX decisions; a mid-size form factor must be considered to display the information;

   UI design is made on UX specifications adding themes, graphics and other elements to personalize the mobile application. UI design has to be in accordance to the layout of the mobile operating system, so a cross-platform mobile application look different on each platform.
3. ***Development*** – represents the application building, using a cross-platform development environment; it starts when the inception stage reached a maturity level;
4. ***Stabilization*** – is made by the Quality Assurance (QA) team getting beta versions of the mobile application which is given to the potential users to get feedback; QA team informs mobile developers for possible changes must be made in the mobile application; stabilization follows the below stage pattern:
   - Prototype – the mobile application has the core functionality or some

parts are working; the proof-of-concept phase is not overcame;
- Alpha – the mobile application has the core functionality implemented completely;
- Beta – the mobile application has a complete implementation; also, a light testing and bug fixing were made;
- Release Candidate – the mobile application is a candidate to be deployed; all functionality is implemented and tested;

5. ***Deployment*** – is made through different ecosystems depending on the platform which the mobile application was design to run for; possible distribution channels are:
   - Application stores – are distribution channels owned by mobile operating providers; they are effective for mobile application distribution and marketing with a small effort by mobile developers;
   - Enterprise deployment – aims internal distribution of the mobile applications within the enterprise interfaces and services; these application are not public through application stores;
   - Ad-Hoc deployment – is a limited deployment to a single or few devices in order to test the mobile application functionality during development stage; it is made via a development environment.

It is not required to be clearly delimited stages during mobile application development lifecycle. Each stage can start before the previous stages to be ended. Thus, the development lifecycle is shorter and development team reduces the development costs.

Also, a software development lifecycle such as Waterfall, Spiral, Agile, etc. can be used to develop mobile applications.

The key elements of the mobile application development lifecycle aim:

- User experience – is improved by developing prototypes of the user interfaces on multiple platforms eventually;
- Testing – must be made on different devices having a wide range of hardware and software features; testing the mobile application on emulators for different platforms is insufficient;
- Maintenance – aims the mobile application update to the last versions of the mobile platforms;
- Moving to the cloud development and deployment – is a new computing paradigm affecting the development process and tools, and the mobile application architecture.

Mobile application development process maps on the traditional development frameworks, but it uses different tools and hardware. Transition from desktop to mobile development can be very difficult because of the mobile application features.

## 4 Conclusions

The paper briefly described the mobile application development process in the context of cross-platform availability and functionality. The mobile engineering faces with mobile devices variety, mobile application features, mobile operating system features, development methodologies and environments, and new paradigms in software development.

Development environments speed up the software development life cycle of the mobile applications. The automation capabilities and plug-ins provide a high quality level for the released mobile applications in a shorter time and they aim the following feature groups:

- Building of mobile application;
- Connection to data storage;
- Usage and performance management;
- Improvement support by community.

It distinguishes the trend to build and use development environments without code editing capability. The mobile application code is automatically generated by IDE as a result of actions in development environment GUI. Other trend is to move the development capabilities in cloud infrastructure where the developer has the all needed tools to build mobile applications on multiple platforms.

**Acknowledgement**
Parts of this paper were presented by the author at "12th International Conference on Informatics in Economy (IE2013) – Education, Research & Business Technologies", Bucharest, Romania, 25 – 28 April 2013.

**References**

[1] J. Doolittle, A. Moohan, J. Simpson and I. Soanes, "Building a Mobile Application Development Framework", *IT@Intel White Paper*, Available: www.intel.com/it, August 2012

[2] I. Ivan, C. Boja and A. Zamfiroiu, "Self-Healing for Mobile Applications", *Journal of Mobile, Embedded and Distributed Systems*, vol. 4, no. 2, pp. 96 – 106, July 2012

[3] P. Pocatilu, M. Doinea and C. Ciurea, "Development of distributed mobile learning systems", in *Proc. of the 9th WSEAS International Conference on Circuits, Systems, Electronics, Control & Signal Processing (CSECS'10)*, Vouliagmeni, Athens, Greece, 2010, pp. 196 – 201

[4] M. Popa, "Frameworks for Mobile Applications Development", in *Proc. of the 12th International Conference on Informatics in Economy*, Bucharest, Romania, April 2013, pp. 101 – 105

[5] M. Popa, M. Doinea and C. Toma, "Java and Symbian M-Application Distribution Frameworks", *Journal of Mobile, Embedded and Distributed Systems*, vol. 1, no. 1, pp. 50 – 59, April 2009

[6] D. Seven, "Cross Platform Mobile Development. Simplified.", Telerik, Inc., October 23, 2012

[7] Xamarin Inc., "Introduction to Mobile Development", Available at: http://docs.xamarin.com/guides/cross-platform/getting_started/introduction_ to_mobile_ development [Jul. 29, 2013]

[8] http://www.appcelerator.com/platform/appcelerator-platform/ [Jul. 29, 2013]

[9] http://www.badrit.com/blog/2013/7/16/sencha-touch-vs-phonegap [Jul. 29, 2013]

[10] http://www.developer.nokia.com/Community/Wiki/Cross_Platform_ Mobile_Architecture [Jul. 29, 2013]

[11] http://devlup.com/mobile/cross-platform-mobile-development-tools/2416/ [Jul. 29, 2013]

[12] http://docs.sencha.com/touch/2.0.2/# [Jul. 29, 2013]

[13] http://docs.xamarin.com/guides/cross-platform/application_fundamentals/ building_cross _platform_applications [Jul. 29, 2013]

[14] https://ec.europa.eu/digital-agenda/node/1505 [Jul. 29, 2013]

[15] http://www.idc.com/getdoc.jsp? Container Id=prUS23946013#.UVLmKRzviSp [Jul. 29, 2013]

[16] http://jquerymobile.com/demos/1.2.1/docs/about/features.html [Jul. 29, 2013]

[17] http://www.maanavan.com/phonegap-application-architecture/ [Jul. 29, 2013]

[18] http://mashable.com/2010/08/11/cross-platform-mobile-development-tools/ [Jul. 29, 2013]

[19] http://www.motorola.com/Business/US-EN/Business+Product+and+Services/Software + and + Applications/RhoMobile+Suite [Jul. 29, 2013]

[20] http://phonegap.com/2012/05/02/phonegap-explained-visually/ [Jul. 29, 2013]

[21] http://readwrite.com/2013/04/16/why-cloud-development-environments-are-better-than-desktop-development#awesm=~ohHMvOIVOjesV8 [Jul. 29, 2013]

[22] http://www.sencha.com/products/touch/features/ [Jul. 29, 2013]

[23] http://www.whitneyland.com/2013/05/why-i-dont-recommend-xamarin-for-mobile-development.html [Jul. 29, 2013]

**Marius POPA** has graduated the Faculty of Cybernetics, Statistics and Economic Informatics in 2002. He holds a PhD diploma in Economic Cybernetics and Statistics. He joined the staff of Academy of Economic Studies, teaching assistant in 2002. Currently, he is Associate Professor in Economic Informatics field and branches within Department of Economic Informatics and Cybernetics at Faculty of Cybernetics, Statistics and Economic Informatics from Bucharest University of Economic Studies. He is the author and co-author of 9 books and over 140 articles in journals and proceedings of national and international conferences, symposiums, workshops in the fields of data quality, software quality, informatics security, collaborative information systems, IT project management, software engineering.